Memory Management Extensions for OpenMP 5.0

2017 CoE Performance Portability Meeting

Denver, Colorado, USA

August 21, 2017

Stephen Olivier

Center for Computing Research, SNL-NM

Sandia National Laboratories

Exceptional service in the national interest

U.S. DEPARTMENT OF ENERGY

National Nuclear Security Administration

Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

SAND2017-8969 PE

Outline

Sandia National Laboratories

- Core features (near completion)
  - OpenMP allocators
  - `allocate` directive and clause
  - `omp_alloc()` and `omp_free()` API routines for C/C++
  - Default allocator
  - `declare alloc` directive and `omp_alloc.h`
- Additional features (less mature)
  - Memory spaces and traits
  - User-defined custom allocators
  - Fallback

OpenMP Allocators

Sandia National Laboratories

- Fundamental concept: object that fulfills allocation requests
- Initially, choice of predefined allocators only:
  - Default
  - High capacity
  - Constant (read-only)
  - High bandwidth
  - Low latency
  - Team local
- Mapping to actual system resources by the implementation
  - E.g., may map all or most to DDR

`allocate` Directive and Clause

Sandia National Laboratories

- Specifies that the OpenMP implementation will perform the memory allocation and indicates which allocator to use

- Directive may be used either by itself or with an associated base language allocation statement
  - Syntax: `#pragma omp allocate(`*`var-list`*`) \`
    `[allocator(`*`predef-allocator`*`)]`

- Clause may be used on `parallel`, `task`, `target`, and worksharing (e.g., loop) directives

- `deallocate` directive for deallocation

C/C++ API Routines

Sandia National Laboratories

- `omp_alloc()` call to allocate memory
  - Specify size and which allocator to use
- `omp_free()` call to deallocate memory
- Strong resemblance to `malloc()` and `free()`

Default Allocator

Sandia National Laboratories

- *Default allocator* used when an allocator is not specified on an allocation directive, clause, or API routine call
- Separate *target default allocator* used only when the `allocate` clause appears on the target directive
  - Allows a different default for non-host allocations
- API routines: `set/get_omp_default_allocator()` and `set/get_omp_target_default_allocator()`
- Environment variables: `OMP_ALLOCATOR` and `OMP_TARGET_ALLOCATOR`

declare alloc

Sandia National Laboratories

- Registers existing allocation functions with OpenMP
- Specifies how to interpret the behavior of the function, its arguments, and its return value:
  - Allocation, reallocation, or deallocation behaviors
  - Size of allocation, pointer to allocation, error code
- Example:
  - Given the function `void* special_alloc(size_t size);`
  - `#pragma omp declare alloc (special_alloc) \ allocate(omp_return) size(omp_args[0])`

`omp_alloc.h` header file

Sandia National Laboratories

- New header file to be delivered by implementations

- Registers popular allocation functions with OpenMP using `declare alloc` directives:
  - `malloc()`
  - `posix_memalign()`
  - `calloc()`
  - `aligned_alloc()`
  - `realloc()`
  - `free()`

Future Features: Memory Spaces

Sandia National Laboratories

- Define memory spaces based on combinations of traits
- Memory space traits span various dimensions, for example:
  - Location (core, socket, device)
  - Capacity and page size
  - Permissions (read, write, both)
  - Bandwidth, latency, or capacity
- Example: device, read-only, low latency memory

Future Features: Custom Allocators

Sandia National Laboratories

- User-defined allocators express desired set of memory space traits
- Implementation finds the memory space that best matches
- Allocators have traits too:
  - Alignment
  - Pinning
  - Shared/exclusive thread model
  - Fallback

Future Features: Fallback

Sandia National Laboratories

- Fallback allocator trait specifies what to do when a memory request cannot be satisfied

- Fallback allocator trait options:
  - Return 0
  - Abort the program
  - Delegate to another specified allocator
  - Try in the default memory space

Other Future Features

Sandia National Laboratories

- Better support for C++
- Explicit optimization for NUMA
- Resource querying
- Special code generation, as required on some new memories
- Static (compile-time) allocator mappings

Other Contributors

Sandia National Laboratories

- Alex Duran (Intel) – *Proposal Lead*
- Christian Terboven (RWTH Aachen) – *OpenMP Affinity Chair*
- Deepak Eachempati and Jeff Sandoval (Cray)
- Kelvin Li and Alex Eichenberger (IBM)
- Alex Rico and Jonathan Beard (ARM)
- John Pennycook, Jason Sewall, Xinmin Tian (Intel)
- Ian Karlin, Tom Scogland, and Bronis de Supinski (LLNL)
- Helen He and Alice Koniges (LBNL)
- Kent Milfeld and Lars Koesterke (TACC)
- <Your name here> -- *Really, please give us feedback!*